ARL-TR-8207 ● Nov 2017

US Army Research Laboratory

Migration of the Three-dimensional Wind Field (3DWF) Model from Linux to Windows and Mobile Platforms

by Giap Huynh and Yansen Wang

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

ARL-TR-8207 ● Nov 2017

US Army Research Laboratory

Migration of the Three-dimensional Wind Field (3DWF) Model from Linux to Windows and Mobile Platforms

by Giap Huynh and Yansen Wang
Computational and Information Sciences Directorate, ARL

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED (From - To)
November 2017	Technical Report	1 October 2016 – 1 October 2017

4. TITLE AND SUBTITLE
Migration of the Three-dimensional Wind Field (3DWF) Model from Linux to Windows and Mobile Platforms

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Giap Huynh and Yansen Wang

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
US Army Research Laboratory
ATTN: RDRL-CIE-M
2800 Powder Mill Road
Adelphi, MD 20783-1197

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TR-8207

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The improved version of the Three-dimensional wind field (3DWF) model, which originally was developed on a Linux operating system, allows the model's initialization either by the user's initial wind inputs and using high-resolution terrain data sets from the Shuttle Radar Topography Mission, or by the use of the wind profiles from the weather research and forecasting mesoscale model outputs. In recognizing the convenience of operating the model and its graphical user interface (GUI) applications on a more compact and popular platform such as Windows, or on mobile devices such as Microsoft's and Google's tablets and smartphones in support of the Soldiers and researchers in isolated areas, the US Army Research Laboratory's microscale modeling team recently made some significant efforts to migrate the 3DWF model from the Linux mainframe computer to the Windows desktop and mobile platforms. This report documents the finished product of the Windows version and the up-to-date efforts as well as the required processes for the transition of the 3DWF model and its GUI to Google's Android platform.

15. SUBJECT TERMS
migration, Windows, mobile platform, Three-dimensional wind field, 3DWF, GUI

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Giap Huynh
Unclassified	Unclassified	Unclassified	UU	32	19b. TELEPHONE NUMBER (Include area code) (301) 394-1156

Contents

List of Figures iv

1. Introduction 1

2. 3DWF Migration Plan 1

3. 3DWF on Windows PC 2

4. 3DWF Graphical User Interface (GUI) on a Windows PC 8

4.1 Running 3DWF Model 9

4.2 3DWF Model Results in netCDF 11

4.3 Morphological Data Generation 16

5. 3DWF on Mobile Platforms 17

5.1 3DWF on Windows Mobile Devices 18

5.2 3DWF Migration to Google's Android Platform 18

6. Conclusion 22

7. References 24

List of Symbols, Abbreviations, and Acronyms 25

Distribution List 26

List of Figures

Fig. 1 3DWF model migration scheme 2
Fig. 2 Operating window of Microsoft's Visual Studio 2010 IDE 3
Fig. 3 Screenshot from Visual Studio 2010 IDE demonstrating how to create a new project 4
Fig. 4 Name a new project 4
Fig. 5 Coding window 5
Fig. 6 Add existing item 5
Fig. 7 Fully imported 3DWF source code programs 6
Fig. 8 Compile selection 7
Fig. 9 Compile result 7
Fig. 10 Build executable code 8
Fig. 11 3DWF GUI's main web page showing Google Maps area NW of Winchester, WV 9
Fig. 12 Resultant wind field in arrow style for a point observation (left) and for WRF data (right) at 30 m above ground level (AGL) 10
Fig. 13 Working window of the Visual Studio 2013 12
Fig. 14 x64 platform setup in Visual Studio 2013 13
Fig. 15 Configuration Manager in Visual Studio 2013 13
Fig. 16 Set up properties for the project 14
Fig. 17 Modifying "Preprocessor Definitions" for the program 14
Fig. 18 Adding netCDF library path for the program 15
Fig. 19 Rasterized images in 3-D Google Earth of buildings (left) and forest canopy (right) 17
Fig. 20 Working window of the NetBeans IDE 19
Fig. 21 Matched wind field results from Fortran90 version (left) and Java version (right) for point observation case (top) and WRF input case (bottom) 22

1. Introduction

The Three-dimensional wind field (3DWF) model is a microscale mass consistent diagnostic model (Wang et al. 2005, 2011) and has recently been improved to use the wind profile data provided by the weather research and forecasting (WRF) (Skamarock et al. 2008) mesoscale model outputs for the model's initialization. The 3DWF model was originally developed on a Linux mainframe computer for fast processing of a very large amount of field test data. However, operating the model on a bulky mainframe computer is not practical in remote areas such as battlefields and field tests where the user may need near-real-time analysis of the field data collection and the model's results. Therefore, it is much more convenient to transition the 3DWF model onto a PC so that internet-accessed graphical user interface (GUI) for the model can be easily developed. Although the model can always be migrated to a Linux operating system PC, the Windows PC is currently much more popular. As a result, there is a need to have the wind model operate either on a compact Windows PC, such as a desktop or laptop, or on a wireless mobile device. In taking advantage of these compact computer and mobile devices, the US Army Research Laboratory's (ARL) Battlefield Environment Division has been making significant efforts to migrate the 3DWF model from a Linux operating system onto a Windows operating system and further onto mobile platforms such as Google's Android or Apple's iOS. This report documents the process to migrate the 3DWF model from Linux to Windows and Android platforms as well as design efforts to create a user-friendly GUI for the updated 3DWF model.

2. 3DWF Migration Plan

Continuous efforts have been made at ARL not only to constantly improve the 3DWF model's performance and to design user-friendly GUI for the model but also to migrate the model to more convenient platforms such as Windows desktop or laptop computers and mobile devices such as tablets and smartphones. Figure 1 illustrates the design scheme and the current status of these efforts. Currently, the fixed-site versions of the 3DWF model on the Linux mainframe computer and on the Windows PC have been successfully completed. Although initial efforts to convert the model's Fortran90 and C code into Java code necessary for Google's Android platform have also been done, the actual deployment of the 3DWF model on a mobile device has not been carried out yet due to unavailability of wireless service at the location where development of the mobile app is being completed.

The deployment of the 3DWF model on mobile platforms will be tested first on Microsoft's Surface Pro tablet once a Wi-Fi service is established.

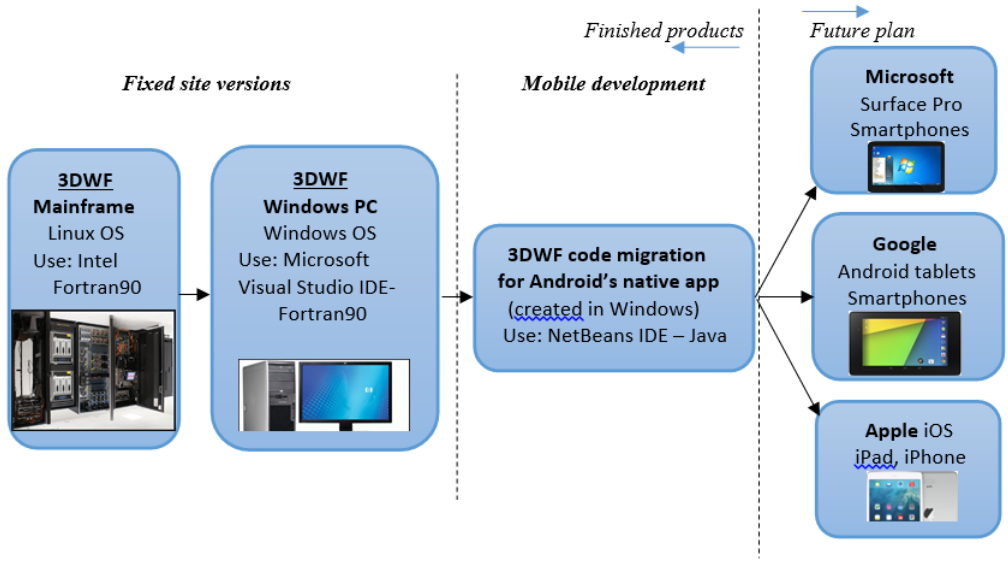

Fig. 1 3DWF model migration scheme

3. 3DWF on Windows PC

In recognizing the convenience of the desktop PC and many free Windows PC software packages from the internet necessary for the migration efforts and for the development of a Google Maps and Google Earth GUI for the 3DWF model, the Microscale Modeling team at ARL has made significant efforts to design a functional and user-friendly GUI that not only helps to run the 3DWF model and produces results in network Common Data Format (netCDF), but also allows optional features to generate morphological data of building and vegetation for wind model users. Since the original version of the 3DWF model was developed on a Linux mainframe computer for faster processing of a large amount of data, operating 3DWF in a bulky machine in areas such as remote battlefields and field tests can be problematic for the Soldier and field test personnel. Therefore, it is more convenient to implement the 3DWF model and the GUI applications on a Windows desktop or laptop computer. In addition, transitioning to a Windows-based PC allows the GUI designer to take advantage of many free software packages available online and developed for PCs, such as netCDF library packages from University Corporation for Atmospheric Research (UCAR) for implementing netCDF translation of the 3DWF model result, and Google Maps application programming interfaces (APIs) for wind field display and for the rasterization of building and forest canopy.

Since the 3DWF's original code was written and compiled in Intel Fortran90 on a Linux operating system, the model's code needs to be recompiled with a Windows-based compiler for the model to run on a Windows PC. This recompilation process can be accomplished by installing a Fortran90 compiler for Windows, such as Microsoft's Visual Studio Integrated Development Environment (IDE) software.

The first step in migrating the 3DWF model to a Windows PC is to obtain a Fortran90 compiler for Windows, such as Microsoft's Visual Studio IDE 2010. This software allows programming and compiling Fortran90 code on a Windows PC. Figure 2 displays the operating window of the Visual Studio 2010 IDE software.

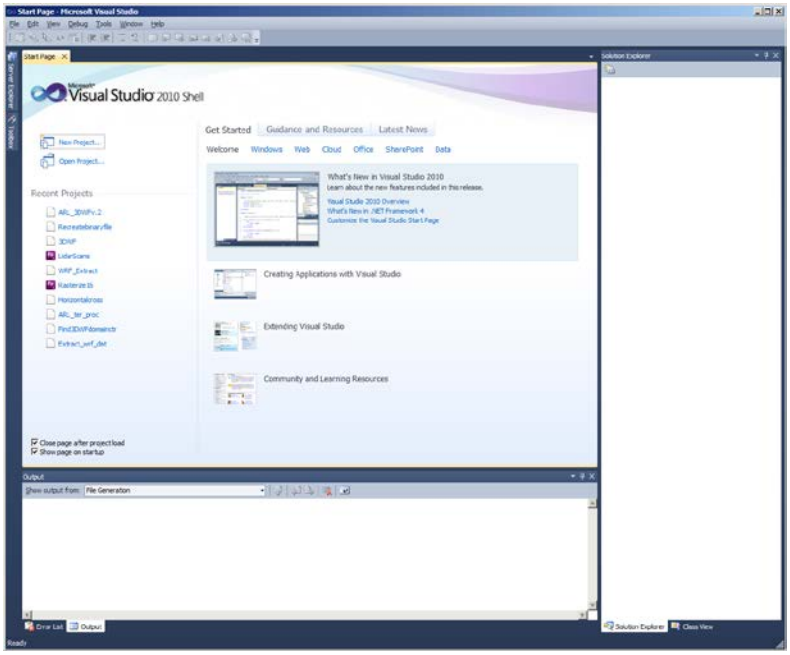

Fig. 2 Operating window of Microsoft's Visual Studio 2010 IDE

The following procedures must be used to import the 3DWF model's main program and its subroutines properly into Visual Studio 2010 IDE.

From the operating window in Fig. 2, click on "File", "New", and then "Project" as shown in Fig. 3 to create a new project.

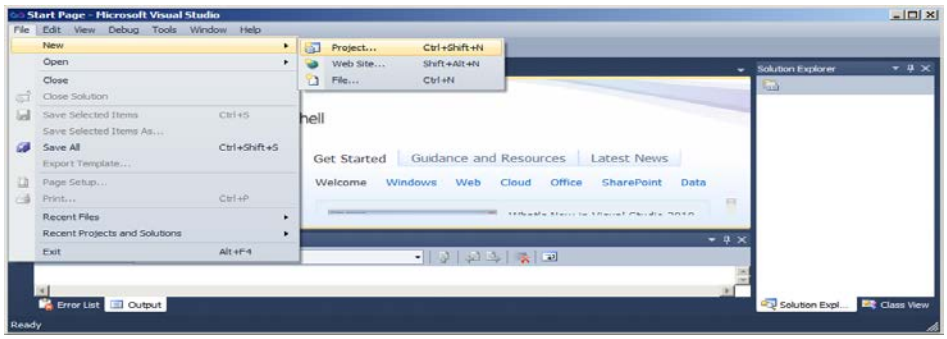

Fig. 3 Screenshot from Visual Studio 2010 IDE demonstrating how to create a new project

After clicking on "Project", a "New Project" window will pop up for entering a new project's name as shown in Fig. 4.

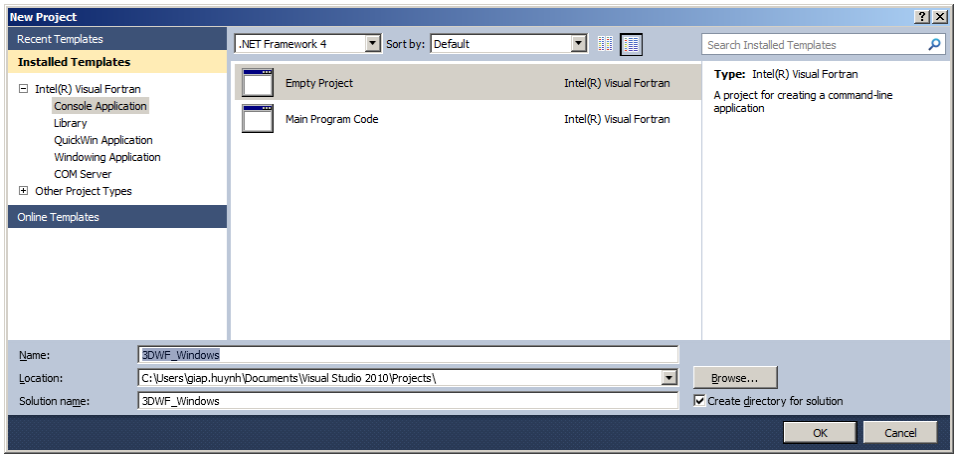

Fig. 4 Name a new project

Click on "Console Application" under the "Recent templates" section, select "Empty Project", and then enter a name (such as "3DWF_Windows" for the above example) for the new project in the slot "Name:" at the bottom of the window. Click OK when done to return to the main window (Fig. 5) at which point it is ready to compose a Fortran90 program.

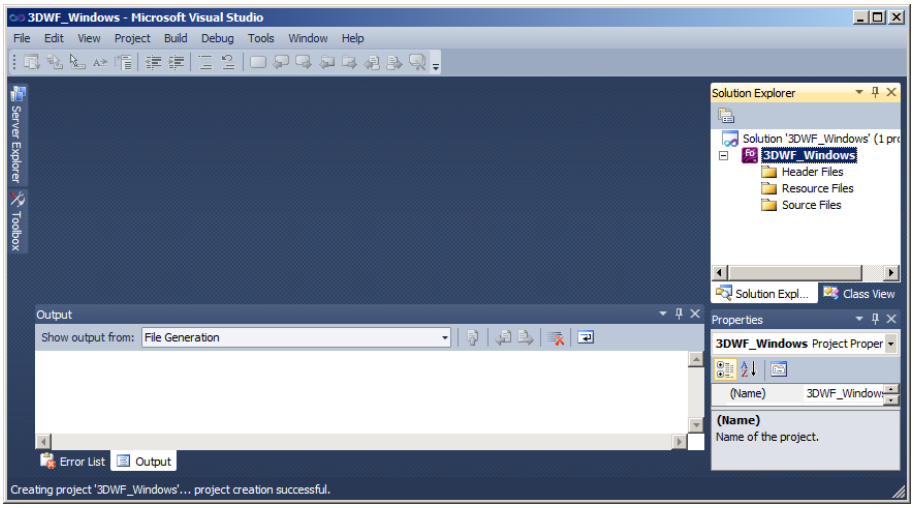

Fig. 5 Coding window

A new Fortran90 program can be composed inside the top-left section of the window. However, since the 3DWF model's Fortran90 source code programs were already created on a Linux mainframe computer, they can be simply imported by clicking "Project" and then selecting "Add Existing Item…" as shown in Fig. 6. A directory window will pop up to allow selecting of an existing Fortran90 program.

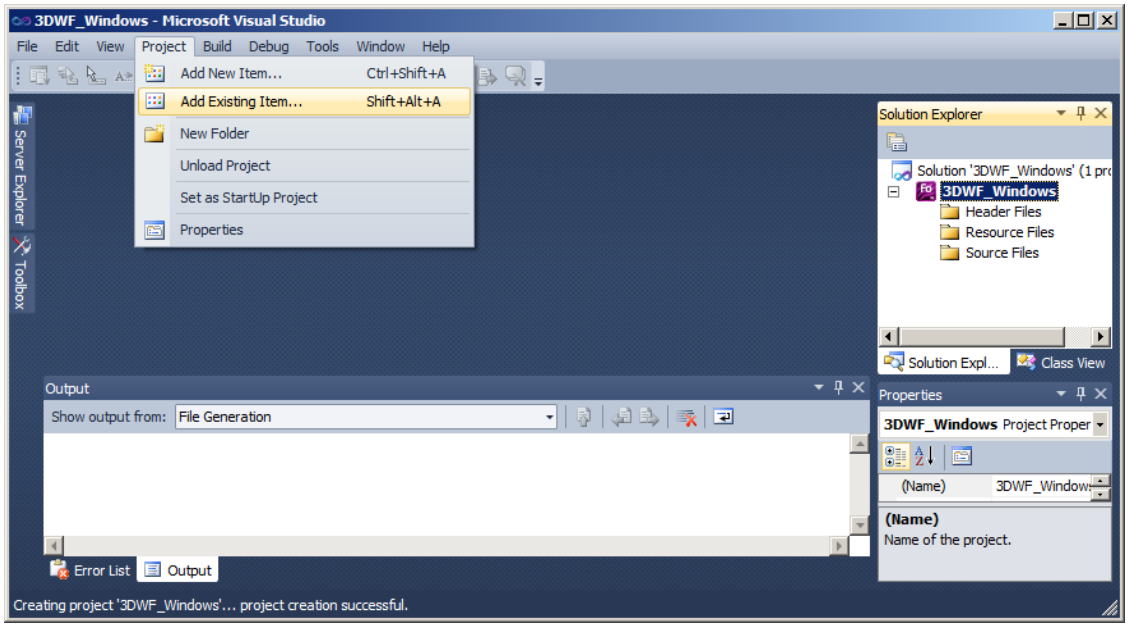

Fig. 6 Add existing item

All imported programs and subroutines can be copied directly from Linux and they will be listed under the "Source Files" in the "Solution Explorer" section at the top right.

For this actual project, the project name is “ARL_3DWFv.2” (instead of “3DWF_Windows”). All imported Fortran90 programs for the updated 3DWF model are listed under the project name “ARL_3DWFv.2”. To display the program’s content, double click on each file to add and display it in the coding area as shown in Fig. 7. The program is now ready for modifying or compiling.

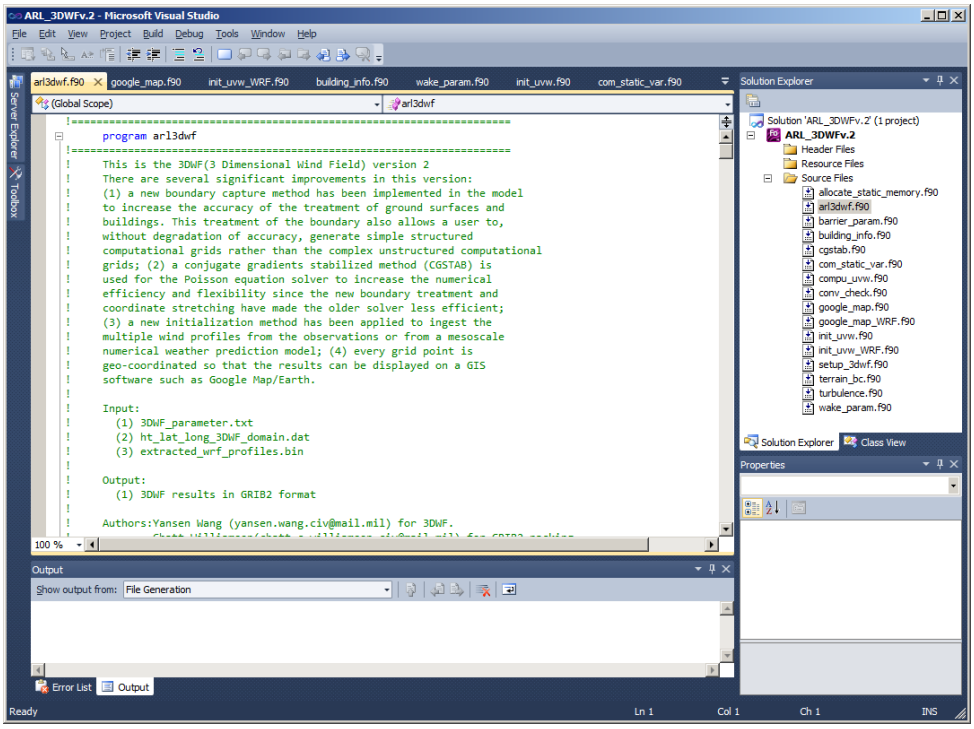

Fig. 7 Fully imported 3DWF source code programs

To compile the main program, click to display the main program “arl3dwf.f90” and then click on “Build” and select “Compile” as shown in Fig. 8.

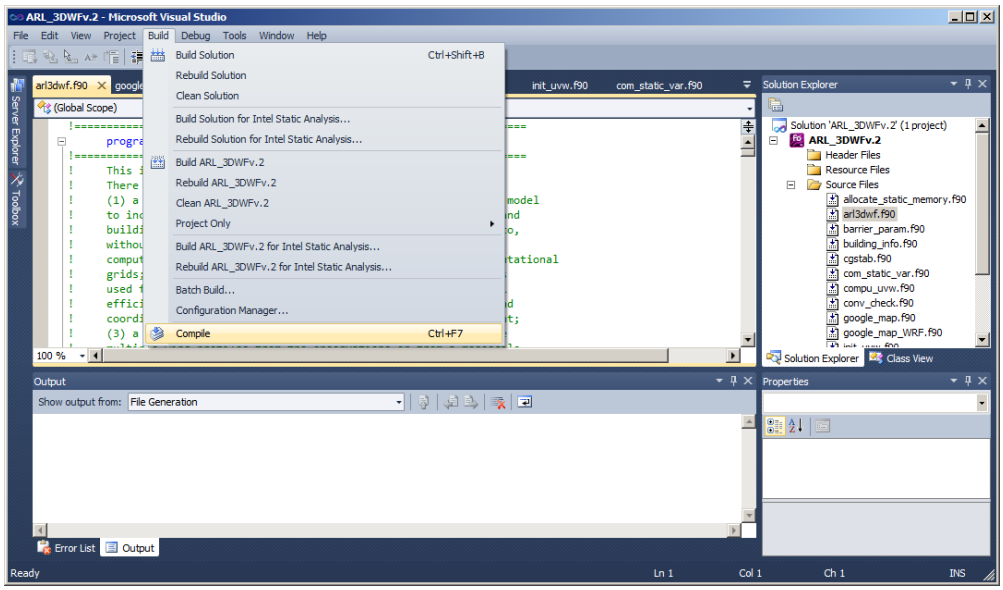

Fig. 8 Compile selection

If successful, the compiling message will indicate “0 error(s)” as displayed in the Output section at the bottom (Fig. 9).

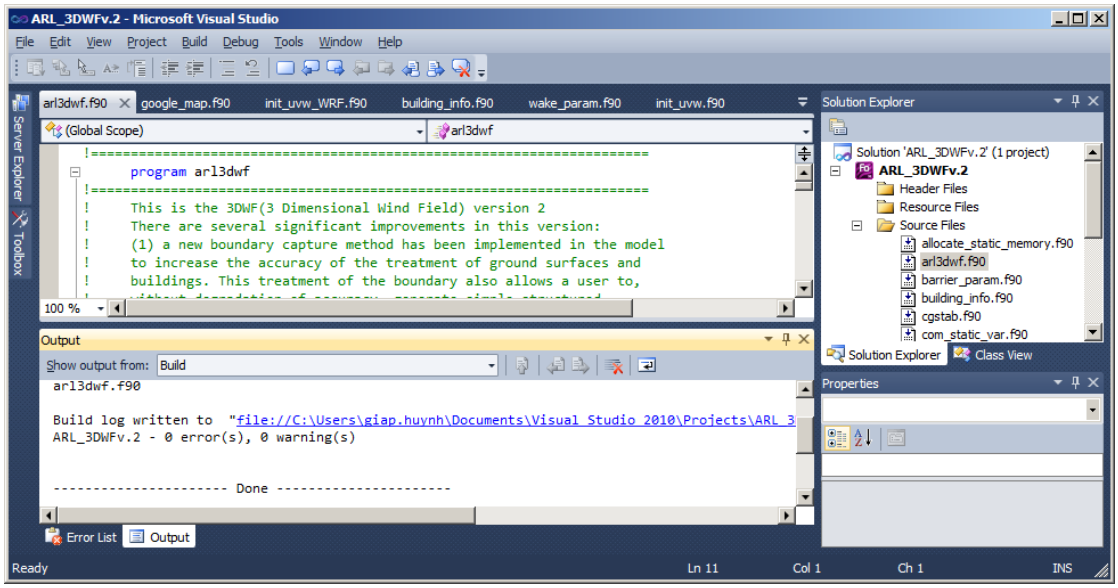

Fig. 9 Compile result

The last step is to build an executable code file with file type “.exe”. Click on “Build” again and then select “Build Solution” to build the executable code file (Fig. 10).

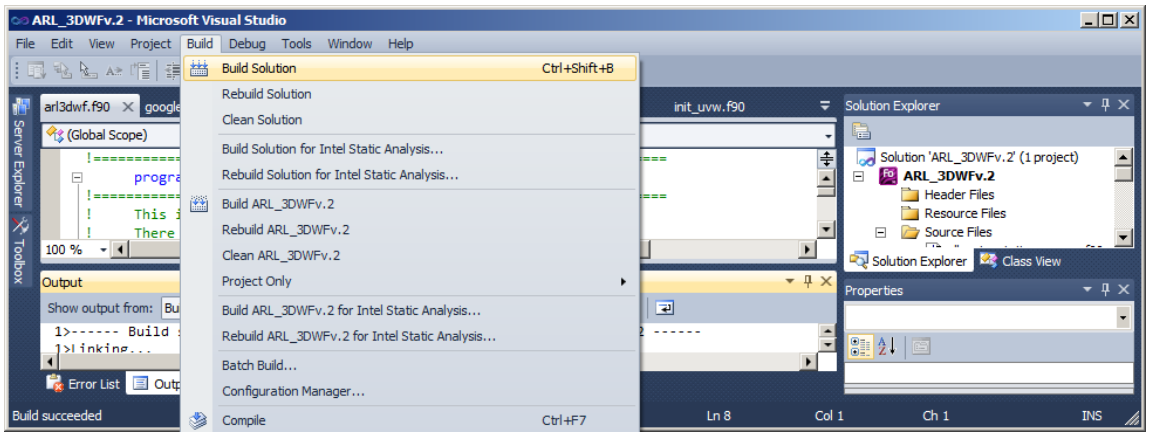

Fig. 10 Build executable code

The executable code file will be generated and stored in the \Debug\ subdirectory following the subdirectory where Visual Studio 2010 was installed. For the 3DWF model, its executable code file "ARL_3DWFv.2.exe" was generated in the following subdirectory:

C:\Users\username\Documents\VisualStudio2010\Projects\ARL_3DWFv.2\ARL_3DWFv.2\Debug\

Similar steps have also been completed for the program for finding terrain domains from the Shuttle Radar Topography Mission (SRTM) data sets. The generated executable code file for this program is named "ARL_ter_proc.exe". These 2 standalone executable files can be copied to run on any GUI application that uses the 3DWF model.

4. 3DWF Graphical User Interface (GUI) on a Windows PC

To assist the user in operating the 3DWF model efficiently on a Windows PC, a user-friendly GUI can be designed in any Windows operating system with internet access via Microsoft's Internet Explorer (IE) web browser. For this particular project, Windows 7 operating system (OS) and the IE 11 browser have been used. Since the GUI needs constant connection to Google Maps over the internet and access to the local C drive and its subdirectories for reading input files and saving the 3DWF model's results, Active X controls permission in IE must be allowed. It is important to understand that enabling Active X controls could put the computer and the network in a vulnerable position. Therefore, it is advised to have network security safeguards in place and operate only in a trusted PC. The GUI's Hypertext Markup Language (HTML) web pages were written in JavaScript to allow the user to run those Fortran90 and C executable programs as mentioned in the previous section to: 1) produce the model results, 2) display results in 2-D Google Maps, 3)

convert the model results into netCDF, 4) generate morphological data for building and forest canopy, and 5) display the building's and forest canopy's polygons in 3-D Google Earth. The 3DWF GUI has also been revised to reflect the 3DWF model's latest improvements. One of those improvements is the option to use the wind profiles data from the WRF model for the 3DWF model's initialization. In summary, the Windows-based 3DWF GUI currently requires the following 5 executable files: ARL_3DWFv.2.exe, ARL_ter_proc.exe, NetCDF_create.exe, ncdump.exe, and Rasterize16.exe. These files must be created and copied to the same subdirectory of the 3DWF GUI prior to its operation. The main web page of the 3DWF GUI is displayed in Fig. 11.

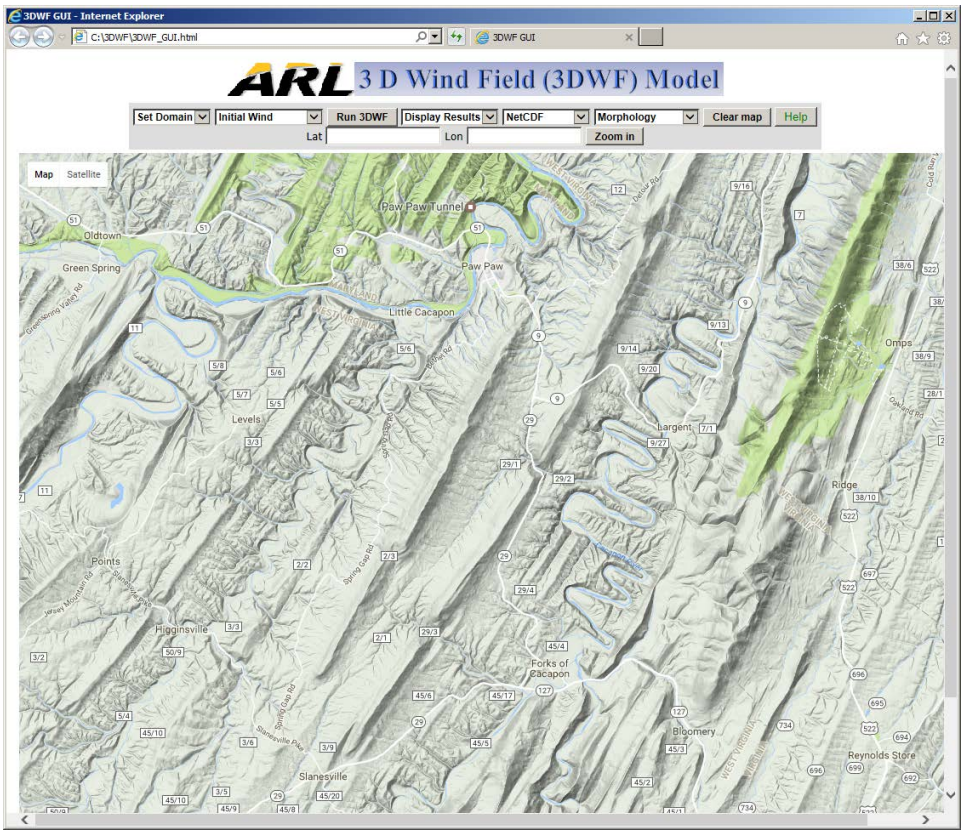

Fig. 11 3DWF GUI's main web page showing Google Maps area NW of Winchester, WV

4.1 Running 3DWF Model

Since the 3DWF GUI displays the wind field results in 2-D Google Maps and also displays the rasterized images of buildings or forest canopies in 3-D Google Earth, it requires internet access during its operation. The GUI's main page was designed with a simple menu at the top that the user can operate from left to right.

From the menu in Fig. 11, the updated 3DWF model allows 2 options for the model initialization: point observation option and the use of the WRF's wind profiles option. For the point observation option, the user starts from "Set Domain". This option allows the user to interactively plot the center of an existing domain on the Google map and to type in the terrain domain's 3-D grid information. The GUI will then automatically execute the file "ARL_ter_proc.exe" to find out whether such a domain is located inside the provided SRTM data sets, and then subsequently to draw its outline. For the WRF data option, it will be started under the "Initial Wind" slot.

For continuing with the point observation case, under the "Initial Wind" slot, select "Point Observation". A small window will pop up, ready for the user to enter basic inputs for the initial wind. For the case of using WRF data, the window will allow the user to enter the WRF domain's grid information.

Once done, the button "Run 3DWF" will execute file "ARL_3DWFv.2.exe" to produce the wind field results.

The last step is to display the 3DWF model's wind field results. The user can select an option under the slot "Display Results" to display the wind field in either wind arrow or wind barb. Figure 12 displays 2 examples of resultant wind field in arrow style over Google Maps for a point observation case (left) and for a case using the WRF's wind profiles (right).

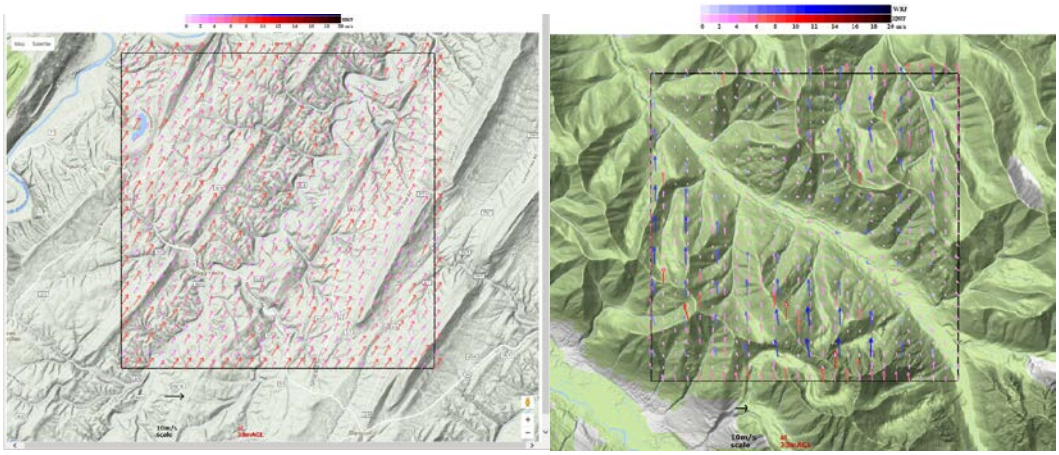

Fig. 12 Resultant wind field in arrow style for a point observation (left) and for WRF data (right) at 30 m above ground level (AGL)

The next 2 optional features produce results in a netCDF file and generate morphological data. These features will be described in detail in sections 4.2 and 4.3. Basically, the "NetCDF" option allows the user to translate the model's result data file from binary to a data file in netCDF (.nc file type). The user can choose

whether to include above ground level (AGL) grid data array of the wind field to be added to the netCDF data file. The GUI also automatically produces a duplicated version in readable text format of the netCDF file. The "Morphology" selection is an optional feature that allows the user to generate morphological data files for a rasterized polygon of a single building or a group of buildings or forest canopy from a given terrain domain. The morphological data generation modules are linked web pages and are not part of the 3DWF model. Therefore, they will be discussed briefly in Section 4.3.

4.2 3DWF Model Results in netCDF

NetCDF is a machine-independent data format that has been widely used in the scientific community as a standard format to support sharing, accessing, and creating scientific data stored in array form. Therefore, it is useful for the 3DWF GUI to also produce results in this format. The Windows-based GUI for the 3DWF model was designed with the ability to translate the original binary result data file into a netCDF data file. To achieve this conversion, the Fortran90 or C conversion program must link to a netCDF library installed on the same computer. UCAR has been developing and distributing various netCDF library packages for different programming languages such as Fortran, C, C++, and Java. This software can be downloaded freely over the internet at the following link:

https://www.unidata.ucar.edu/downloads/netcdf/index.jsp

Since the netCDF library package for Fortran90 was not yet available at the time when the 3DWF GUI was being developed, the netCDF library package for C language named as "netCDF 4.3.2" was used instead. As a result, the programs to translate the model results into netCDF were written in C language via Microsoft's Visual Studio 2013 for Windows users. The procedures to create and compile a program in Visual Studio 2013 are similar to those in the Visual Studio 2010 as described previously, except for some additional required configurations. An example of a Visual Studio 2013 working window is displayed in Fig. 13.

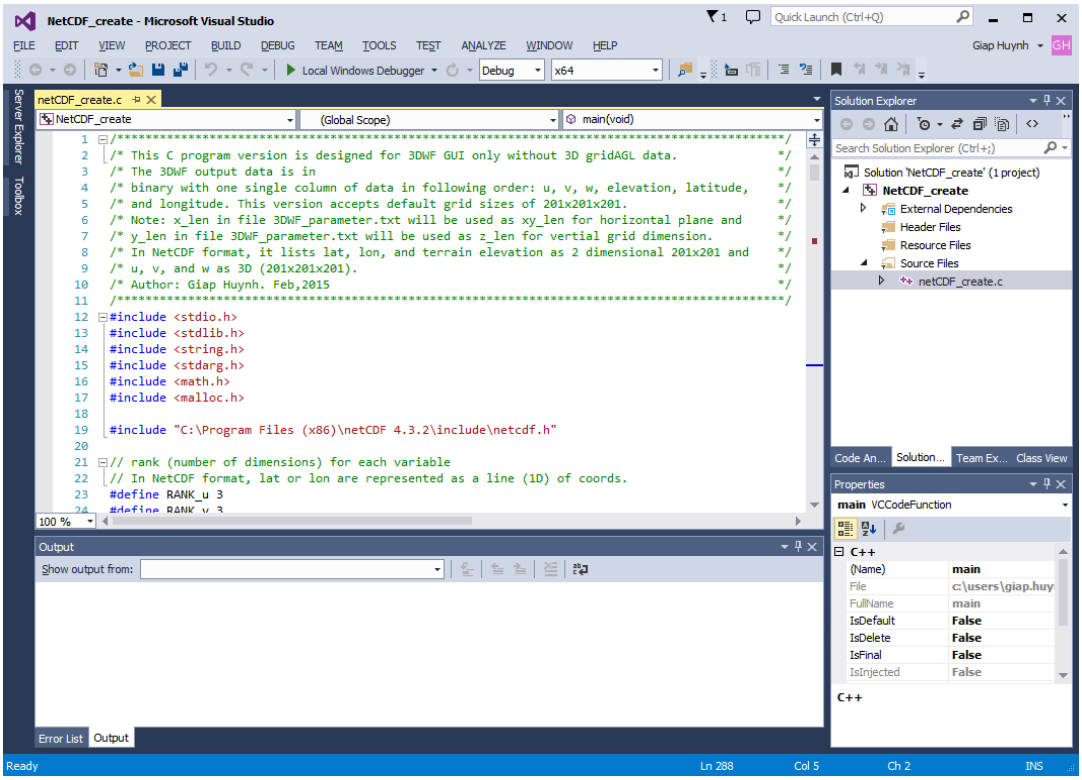

Fig. 13 Working window of the Visual Studio 2013

The following are additional setup steps before compilation:

1) It is important to match the netCDF for C library package's bit version with the Visual Studio 2013's compile bit platform. Since the downloaded netCDF package is a 64-bit version, Visual Studio 2013 must also be set up to compile a C program in the x64 platform. From the "Debug" slot at the center top of the Visual Studio 2013 working window, click the down arrow to select "Configuration Manager…" as shown in Fig. 14.

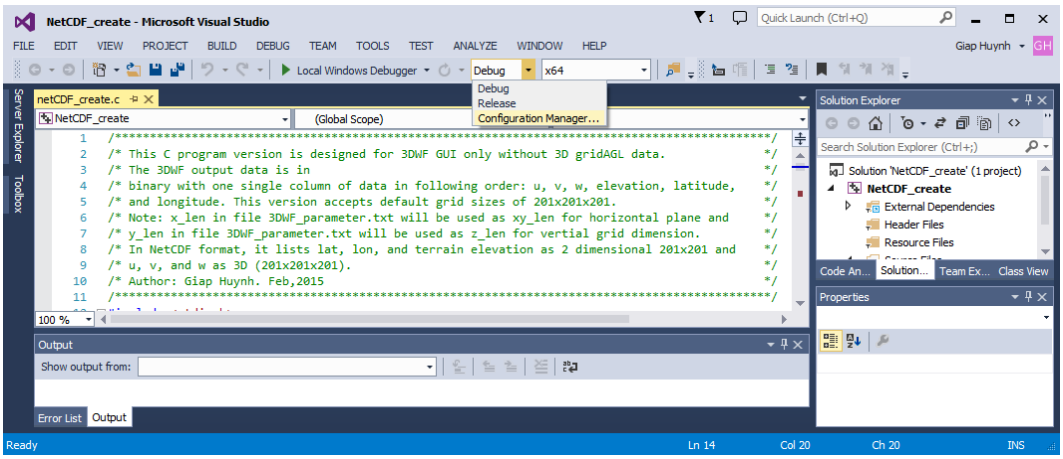

Fig. 14 x64 platform setup in Visual Studio 2013

The "Configuration Manager" window will pop up as shown in Fig. 15.

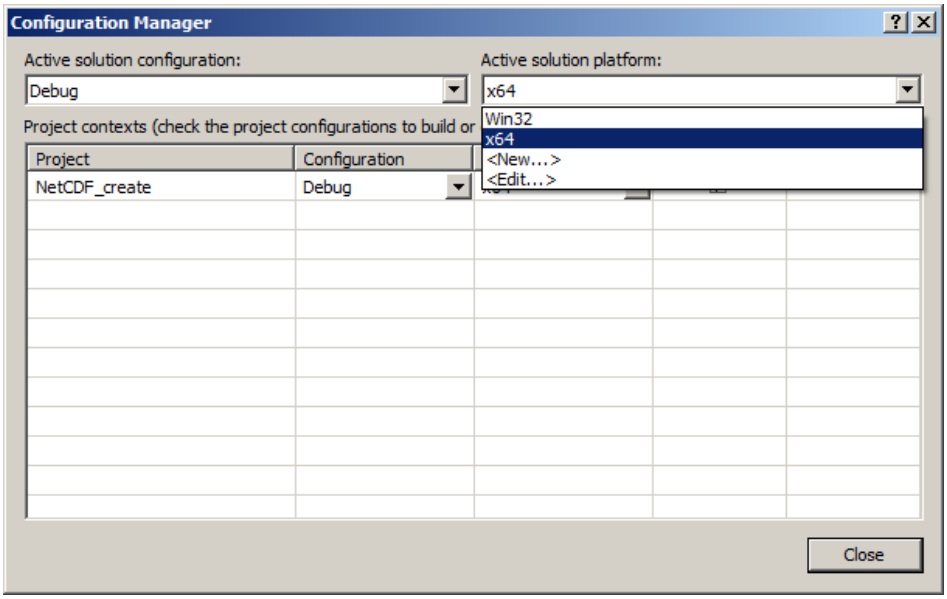

Fig. 15 Configuration Manager in Visual Studio 2013

From the "Active solution platform:" option, click the down arrow to select x64 and then click "Close".

2) Next, click on the "PROJECT" option from the main window and select the project properties. For example, "NetCDF_create Properties…" is the project properties for the "netCDF_create.c" program (Fig. 16).

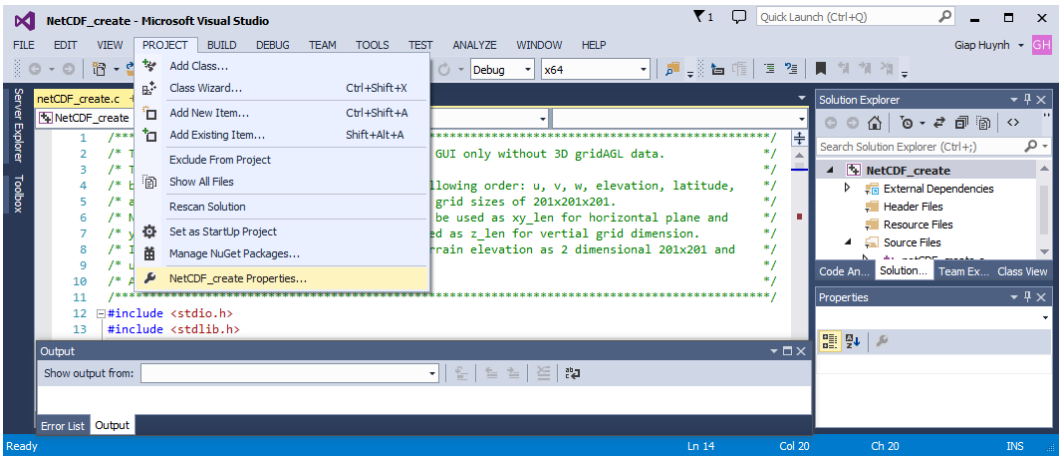

Fig. 16 Set up properties for the project

A property window will appear as shown in Fig. 17. Next, select “Configuration Properties”, “C/C++”, and then “Preprocessor”. Edit “Preprocessor Definitions” option at the top of the list by adding the error message “_CRT_SECURE_NO_WARNINGS” after “WIND32”.

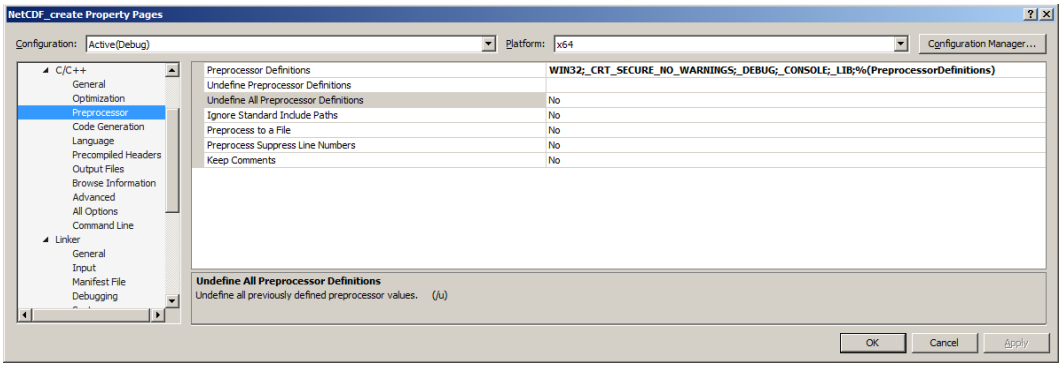

Fig. 17 Modifying “Preprocessor Definitions” for the program

3) Continue down the list in the “Configuration Properties” sub-window, click on “Linker”, and then select “Input” to allow adding paths in the slot for “Additional Dependencies” (Fig. 18). Append the following netCDF library path to the end of the existing path in the “Additional Dependencies” slot:

C:\Program Files (x86)\netCDF 4.3.2\lib\netcdf.lib

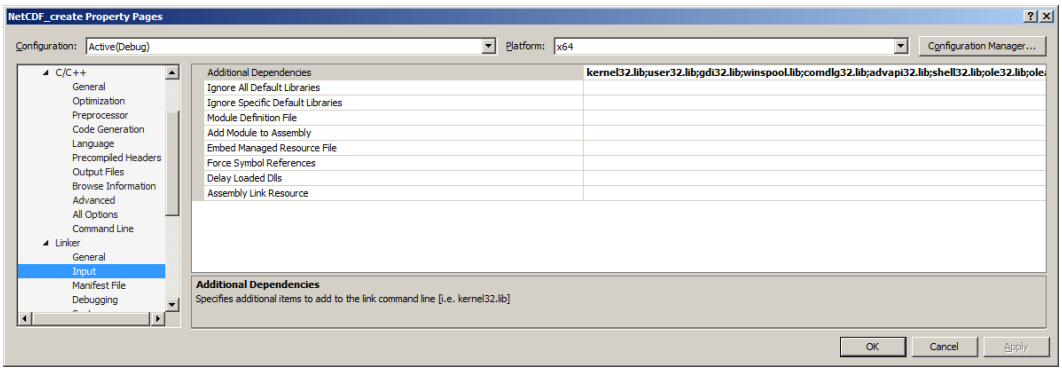

Fig. 18 Adding netCDF library path for the program

4) Make sure that these 4 files are copied and saved in the same subdirectory as the executable code's subdirectory: "netcdf.dll", "hdf5.dll", "hdf5_hl.dll", and "zlib.dll". For the "NetCDF_create" program example, they should all be under this subdirectory:

C:\Users\giap.huynh\Documents\Visual Studio 2013\Projects\NetCDF_create\x64\Debug\

Notice that while 3 files ("hdf5.dll", "hdf5_hl.dll", and "zlib.dll") are generated during the program code compilation, file "netcdf.dll" must be copied from the netCDF library:

C:\Program Files (x86)\netCDF 4.3.2\bin

To convert a netCDF file (file type .nc) to a readable text file, copy file "ncdump.exe" (generated during netCDF installation) to the subdirectory where it will be executed (usually 3DWF GUI's subdirectory). The text version of the beginning of a netCDF file is shown in the following example:

```
netcdf \3DWF_NetCDF_withoutAGLgrid {
dimensions:
        level = 201 ;
        latitude = 201 ;
        longitude = 201 ;
variables:
        float latitude(latitude, longitude) ;
                latitude:units = "degrees_north" ;
        float longitude(latitude, longitude) ;
                longitude:units = "degrees_east" ;
        float ter_elev(latitude, longitude) ;
                ter_elev:units = "m" ;
        float u(level, latitude, longitude) ;
                u:units = "m/s" ;
        float v(level, latitude, longitude) ;
                v:units = "m/s" ;
        float w(level, latitude, longitude) ;
                w:units = "m/s" ;
```

data:
latitude =
39.39566, 39.39566, 39.39566, 39.39566, 39.39566, 39.39566, 39.39566,
39.39566, 39.39566, 39.39566, 39.39566, 39.39566, 39.39566, 39.39566,
39.39567, 39.39567, 39.39567, 39.39567, 39.39567, 39.39567, 39.39567,
……..

Once done, the executable code "NetCDF_create.exe" is produced. This standalone file will be used by the 3DWF GUI to generate the model results in netCDF with 2 options, with and without AGL grid's data arrays added to the file. Notice that 4 files ("netcdf.dll", "hdf5.dll", "hdf5_hl.dll", and "zlib.dll") must follow the executable file "NetCDF_create.exe" if this file is moved and executed in another subdirectory.

4.3 Morphological Data Generation

It is also useful for the 3DWF GUI to have a capability to generate morphological data files of buildings and the forest canopy based on Google's satellite map. Originally, this feature was successfully designed based on the 3-D Google Earth API to allow the user to locate and draw polygons of buildings and forest canopies over 3-D Google Earth's satellite map (Huynh et al. 2010). However, since Google deprecated the Google Earth API plug-in due to security issues, Google Maps API was successfully used to replace the Google Earth API on all applications. Although Google Earth API, which allowed the user the ability to plot and draw to rasterize or digitize a building over satellite image, is no longer available, the GUI still needs Google Earth display-only software to display the 3-D resultant image of buildings and forest canopies as shown below in Fig. 19. The program for these modules was written in Fortran90 and compiled via Visual Studio 2010. The executable code for these features is named "Rasterize16.exe" and allows the user to rasterize buildings or forest canopies with up to 16 sides and then produces morphological data files for them. Click on the "Morphology" slot to select 1 of the 2 options, either "Building polygon" for a single building or "Canopy polygon" for a group of buildings or a forest canopy. This feature will open a related linked web page to generate morphological data files and display the rasterized object in 3-D Google Earth. Figure 19 displays examples of the 2 options in 3-D Google Earth.

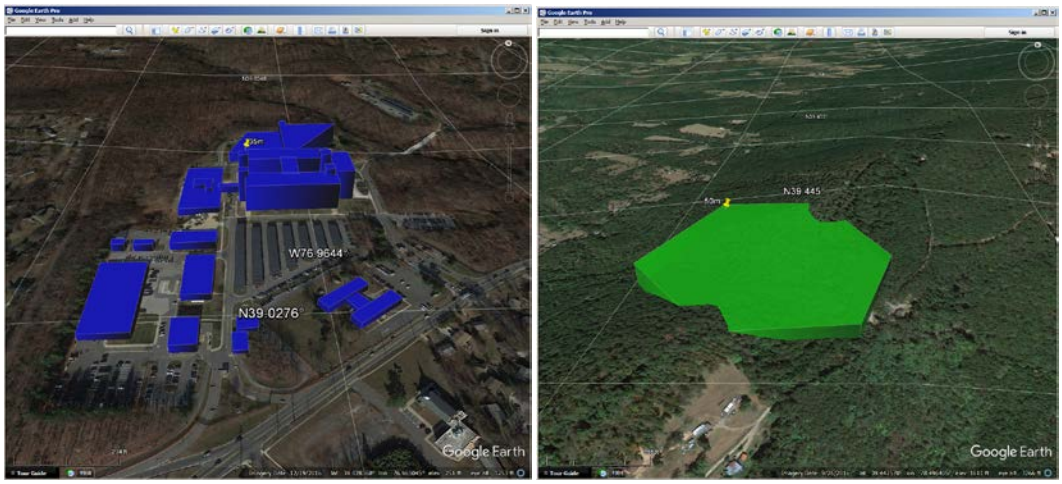

Fig. 19 Rasterized images in 3-D Google Earth of buildings (left) and forest canopy (right)

5. 3DWF on Mobile Platforms

Although migrating the 3DWF model from a mainframe computer to a PC provided some conveniences, the model is still operating at a fixed site. Desktop or laptop PCs are still bulky and heavy for the Soldiers and field test personnel who often need to move around and usually operate in isolated areas such as remote battlefields and field test sites. Therefore, ARL's Battlefield Environment Division has also been making significant efforts to transition the 3DWF model not only onto PCs but also further onto mobile devices such as tablets and smartphones. Although wirelessly operating the 3DWF GUI on tablets and cellphones can be challenging due to a number of security concerns, limited memory, and small display screens, it is still useful to initiate this new development as the mobile devices become more powerful and popular in the future.

Currently there are 3 main operating systems for the tablet and smartphone market: Microsoft's Windows, Google's Android, and Apple's iOS. Of these 3 operating systems, Google's Android and Apple's iOS currently seem to be more widely used than Microsoft's Windows, based on the annual sold figures of their tablets and smartphones. Unlike Microsoft's Windows, Google's and Apple's mobile platforms do not support Fortran. Since Google's mobile platform uses Java and Apple's mobile platform adopts Objective C or Swift, it makes sense in the long term to also start making efforts to transition the 3DWF model onto these 2 popular mobile platforms. For this technical report, the initial focuses are on the Windows mobile platform and Google's Android platform. Transitioning to Apple's iOS platform will be looked at thereafter.

5.1 3DWF on Windows Mobile Devices

Since Windows PC and Microsoft's mobile devices share the same Windows operating system and since the 3DWF model and its GUI have already been migrated successfully to the Windows desktop PC, the 3DWF model and its GUI programs can be uploaded to any Microsoft tablets, such as the Surface Pro tablet, or to Windows-based smartphones, such as the Lumia smartphone. Unlike Android mobile devices, which require their applications to be designed exclusively on Java code, Windows PCs and Windows mobile devices can run applications written in either Fortran90 or Java code. Due to network security issues at ARL and the unavailability of a Wi-Fi service, the actual deployment of the 3DWF model on Windows-based mobile devices, such as on the Surface Pro tablet, will be done as soon as those issues are resolved.

5.2 3DWF Migration to Google's Android Platform

As mentioned previously, since Google's Android OS adopted Java code as its native language and does not run Fortran executable files, the Fortran90 programs written for the 3DWF model on Windows must be converted into Java code. To accomplish this task, a Windows-based Java IDE software package must be installed on the Windows desktop PC to allow Java coding and compilation. Currently, there is a user-friendly IDE package called NetBeans 8.0.2 which is free for downloading from the internet. This software was chosen and installed for this Java code conversion effort. Figure 20 displays the working window of the NetBeans 8.0.2.

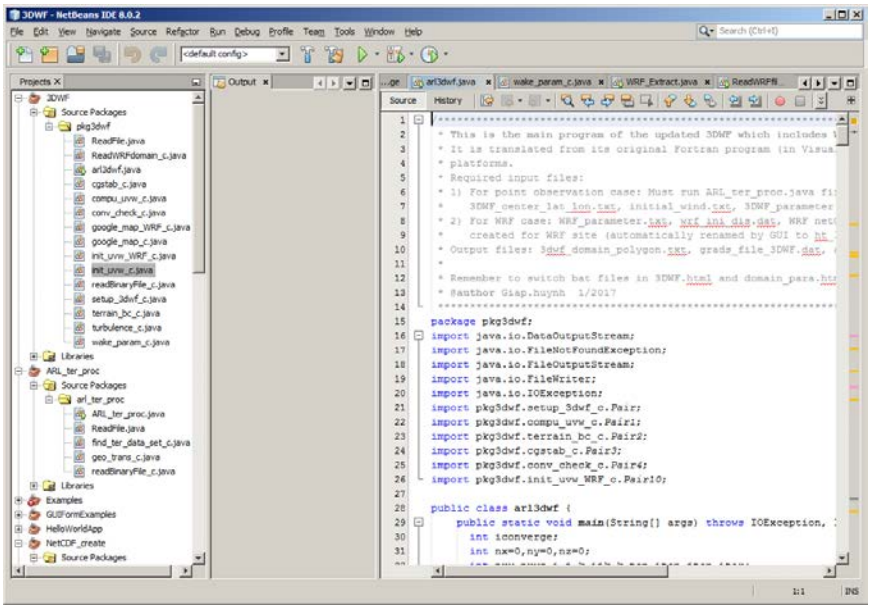

Fig. 20 Working window of the NetBeans IDE

The Java code conversion efforts have been done for all 3 following separate features that were previously written in Fortran and C languages: the 3DWF model programs, the programs to convert the model results to netCDF, and the morphological data generation modules. All executable code files ".jar" generated by the following Java programs must also be copied to the subdirectory C:\3DWF\.

For the 3DWF model programs:

From the user-friendly operating window of the NetBeans IDE above, the user can select "File" and "New Project" to start a project. A Java program is composed inside the "Source" window to the right while the "Projects" window on the left displays the project's programs and subroutine files. The "Output" window in the middle is reserved for the progress messages during running or compiling. During the compilation, the software automatically detects whether additional library imports are needed or if the variables must be initialized. Alt-Enter key combination can be used to view additional suggestions for fixing errors. Once coding is completed, either left click on the "Run" option and then select "Run File", or right click on the main program under the "Projects" window and select "Run File" to compile and run the program.

The translation from Fortran90 to Java code has been done in a straightforward manner to keep the Java programs as close as possible to their original Fortran90 version. Therefore, all file names, subroutine names, and variables are unchanged. Notice that this effort is based on the basic 3DWF GUI version and does not include

the wake parameterization around the building subroutine. The following are a list of all Java files for the 3DWF model.

Java files for SRTM terrain data processing:

ARL_ter_proc.java, find_ter_data_set_c.java, geo_trans_c.java, ReadFile.java, and readBinaryFile_c.java

Java files for the 3DWF model:

arl3dwf.java, cgstab_c.java, compu_uvw_c.java, conv_check_c.java, google_map_WRF_c.java, google_map_c.java, init_uvw_WRF_c.java, init_uvw_c.java, setup_3dwf_c.java, terrain_bc_c.java, turbulence_c.java, ReadFile.java, readBinaryFile.java, and ReadWRFdomain_c.java

Notice that files "ReadFile.java", "readBinaryFile_c.java", and "ReadWRFdomain_c.java" are additional subroutines to read text (the first file) and binary (the last 2 files) data files. To produce a standalone executable file ".jar" to be used in a GUI, select "Build Main Project" or "Clean and Build Main Project" to generate the file. For example, if the main Java program file is "3DWF.java", then its executable file, "3DWF.jar", is generated under this subdirectory:

C:\Users\giap.huynh\Documents\NetBeansProjects\3DWF\dist\

For the netCDF data translation:

For distribution purposes, the 3DWF GUI is also implemented with a feature to allow the translation of the model's resulting data file from binary to netCDF format. Therefore, a netCDF for Java library package must also be installed on the Windows desktop PC. The package is free for downloading from UCAR's website. Some setup steps must be done in NetBeans to allow the Java program to link to the installed netCDF library before the code can be compiled. Under the project name inside the "Projects" window of NetBeans, double click on "Libraries" to display available linked libraries. To add a new linked library, right click on "Libraries" and select "Properties" to pop up the "Project Properties" window. From this property window, use Browse to locate the directory path that leads to the location of the netCDF for Java's libraries. For example:

C:\Program Files (x86)\netcdfAll-4.6.6.\

Next, click on "Add JAR/Folder" and then click a check mark on "Absolute Path". Click "Open" to view the path to the libraries and then click "OK" when done.

During the running of the Java code to produce netCDF data, the user may see some error messages showing up. In such a case, an additional library named "slf4j-nop-1.7.22.jar" can be downloaded freely from the internet and can be installed to

remove these error messages. All linked libraries for this project are listed under "Libraries".

Notice that the netCDF data file can be produced either from the 3DWF GUI via a batch file to execute the file "NetCDF_create.jar" or from running the standalone Java code inside the NetBeans operating page. Therefore, if for some reason the netCDF data file cannot be produced through the 3DWF GUI, the user can always choose to independently run the "NetCDF_create.jar" program inside NetBeans to produce the netCDF data file. Java code programs for the netCDF feature are "NetCDF_create.java", "ReadFile.java", and "readBinaryFile_c.java".

For morphological data generation

The Fortran90 code for morphological data generation must also be translated into Java using the same NetBeans software. A complete listing of Java files for this feature are: "Rasterize16.java", "ReadFile.java", "readBinaryFile_c.java", and "loctempgrid_c.java, and rasterize_func_c.java".

In summary, after all programs and subroutine files are compiled, the following are all generated standalone Java executable code files required for the 3DWF GUI application:

ARL_ter_proc.jar	(Process SRTM terrain data to create a domain for Point Observation case)
3DWF.jar	(Executable file of the 3DWF model)
NetCDF_create.jar	(Executable file to produce the model's result in netCDF format)
Rasterize16.jar	(Produces morphological data of building or forest canopy)

Although actual deployment of the 3DWF model and its GUI on a Google's Android OS device has not been carried out, their complete Java code programs have been tested successfully on a Windows desktop PC (since Windows PCs can also execute Java executable code). Test runs with actual provided data were performed on a Windows PC for both Fortran90 and Java versions for comparison. The matching results (Fig. 21) on both versions proved that the conversion effort into Java code was successful and the code is ready for the 3DWF model's deployment onto a Windows or an Android mobile device.

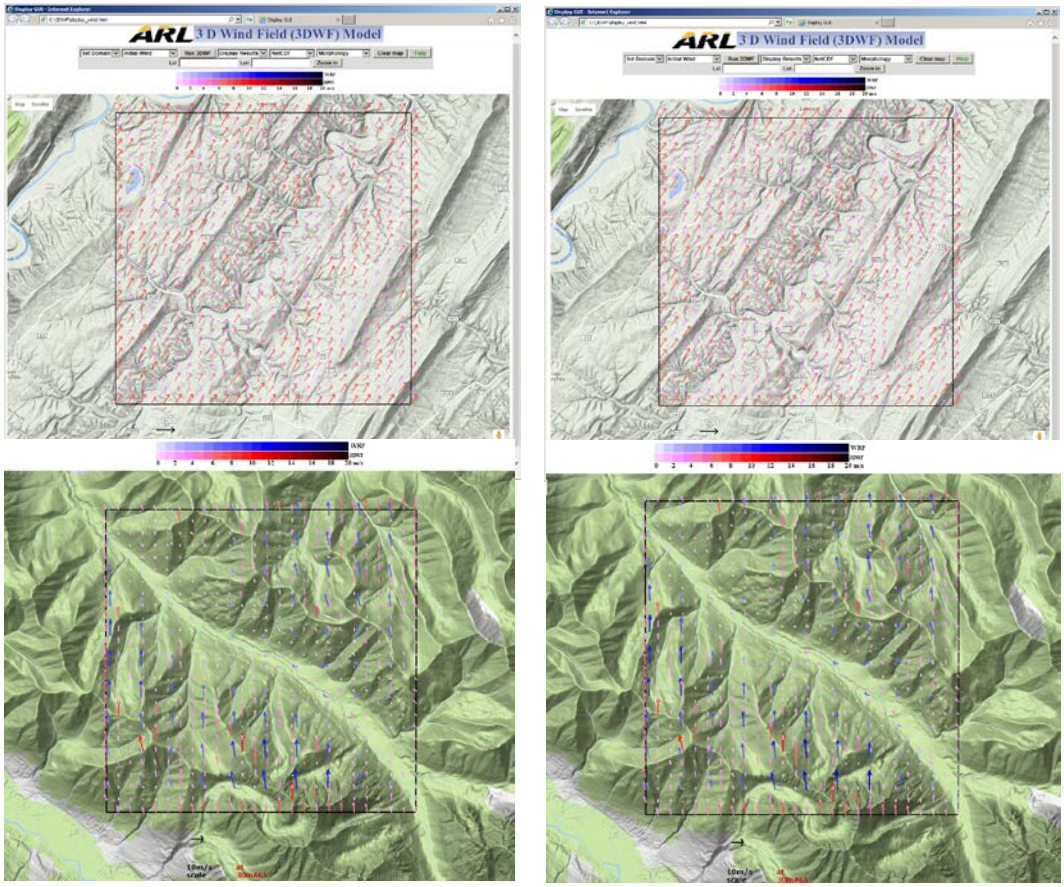

Fig. 21 Matched wind field results from Fortran90 version (left) and Java version (right) for point observation case (top) and WRF input case (bottom)

6. Conclusion

Significant efforts have recently been made at ARL's Battlefield Environment Division to transition the improved 3DWF model from a Linux mainframe computer to a Windows desktop PC and mobile platforms such as Microsoft's Windows tablets and Google's Android tablets and smartphones. For the Windows PC, the 3DWF and its customized GUI package were successfully migrated and tested on a desktop PC with access to wired internet. This successful effort not only helps potential customers to install the 3DWF package on widely used compact PCs such as Windows desktops and laptops but also allows the model's user to quickly run the model through the use of a well-designed and user-friendly GUI. For mobile platforms, preliminary efforts have been focused on migrating the 3DWF onto a Windows tablet and a Google mobile device. The transition of the 3DWF model and its GUI onto a Windows tablet is similar to its transition to a Windows PC since they both share the same Windows OS. As for the transition onto Google's Android OS, all programs for the 3DWF model and for its GUI's

features have been successfully converted into Java and performed successfully on a Windows desktop PC using real input data. The results were also compared and matched with the results from the Fortran90 version to ensure that the 3DWF model and its GUI will perform properly after future deployments on either a Windows mobile device or an Android mobile device.

7. References

Google Earth. Mountain View (CA): Google [accessed 2017 Nov 2]. https://www.google.com/earth.

Google Maps APIs. Mountain View (CA): Google [accessed 2017 Nov 2]. https://developers.google.com/maps/documentation/javascript.

Huynh G, Wang Y, Williamson C. Building and vegetation rasterization for the three-dimensional wind field (3DWF) model. Aberdeen Proving Ground (MD): Army Research Laboratory (US); 2010. Report No.: ARL-TR-5419.

Skamarock WC, Klemp JB, Dudhia J, Gill DO, Barker DM, Duda MG, Huang X-Y, Wang W, Powers JG. A description of the advanced research WRF version 3. NCAR Tech Note: NCAR/TN-475+STR; 2008 [accessed 2017 Nov 2]. http://dx.doi.org/10.5065/D68S4MVH.

Wang Y, Williamson C, Garvey D, Chang S, Cogan J. Application of a multigrid method to a mass consistent diagnostic wind model. J Appl Meteorol. 2005;44:1078–1089.

Wang Y, Williamson C, Huynh G. Three-dimensional wind field (3DWF) model for AFWA's applications: a customer project report. 2011. p. 6–8.

List of Symbols, Abbreviations, and Acronyms

2-D	2-dimensional
3-D	3-dimensional
3DWF	Three-dimensional wind field
AGL	above ground level
API	application programming interface
ARL	US Army Research Laboratory
GUI	graphical user interface
HTML	Hypertext Markup Language
IDE	Integrated Development Environment
IE	Internet Explorer
netCDF	network Common Data Format
NW	northwest
OS	operating system
PC	personal computer
SRTM	Shuttle Radar Topographic Mission
UCAR	University Corporation for Atmospheric Research
WRF	weather research and forecasting

1 (PDF)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA
2 (PDF)	DIR ARL IMAL HRA RECORDS MGMT RDRL DCL TECH LIB
1 (PDF)	GOVT PRINTG OFC A MALHOTRA
10 (5 HC, 5 PDF)	ARL RDRL CIE P CLARK RDRL CIE D G VAUCHER RDRL CIE M G HUYNH (5 HC) B MACCALL Y WANG RDRL WML B J CIEZAK-JENKINS